

BAB III

RANCANG BANGUN

A. Tujuan Rancang Bangun

Penelitian ini bertujuan untuk dapat merancang *stand* kelistrikan, menentukan komponen yang akan digunakan pada pembuatan alat ini serta penempatan komponen - komponen yang ada didalamnya selain itu juga untuk membangun suatu sistem yang dapat menghidupkan dan mematikan sistem *starter*, sistem penerangan (lampu utama dan lampu kecil) serta sistem *signal* (lampu *sign* dan *hazard*) dengan menggunakan sensor suara.

B. Waktu dan Tempat Rancang Bangun

Pelaksanaan penelitian ini dilakukan di Laboratorium Teknik Sepeda Motor dan Laboratorium Teknik Elektronika Industri, SMK Negeri 2 Kota Bekasi. Adapun kegiatan ini dilaksanakan pada bulan Juni 2015 sampai dengan selesai.

C. Alat Dan Bahan

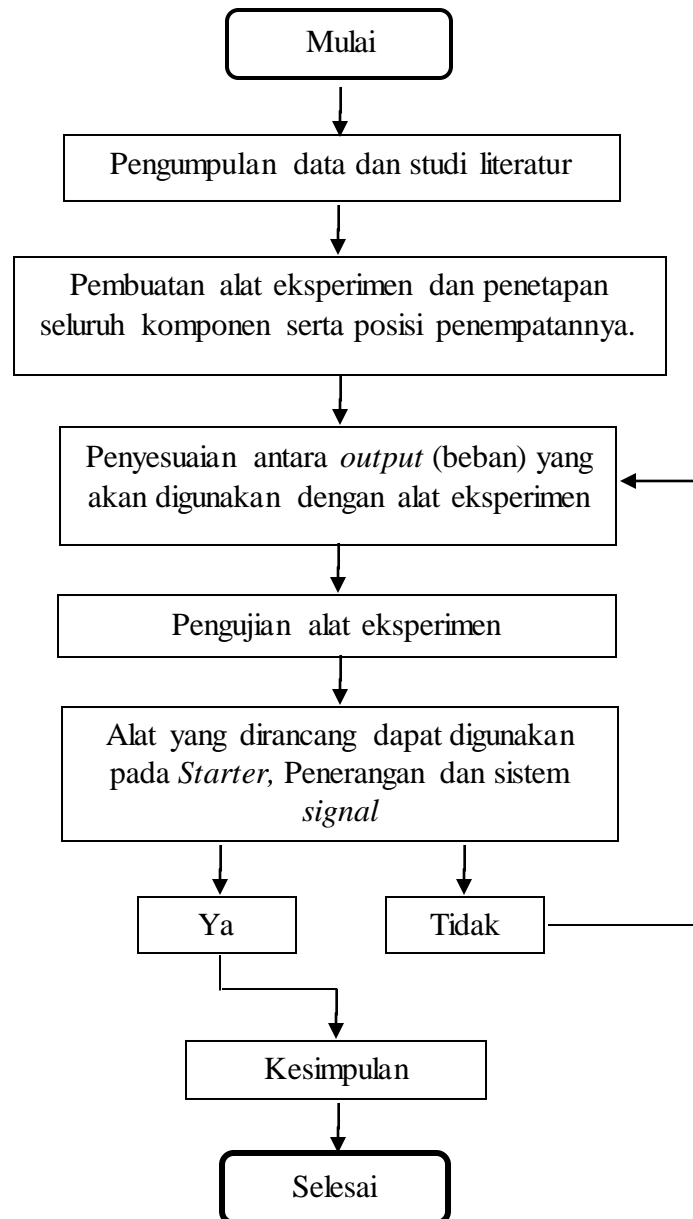
Alat dan bahan yang digunakan pada rancang bangun ini adalah:

1. Sensor Suara *EasyVR*
2. Mikrokontroler
3. DI-Smart AVR 16 *System*
4. *LCD Board* 2x16

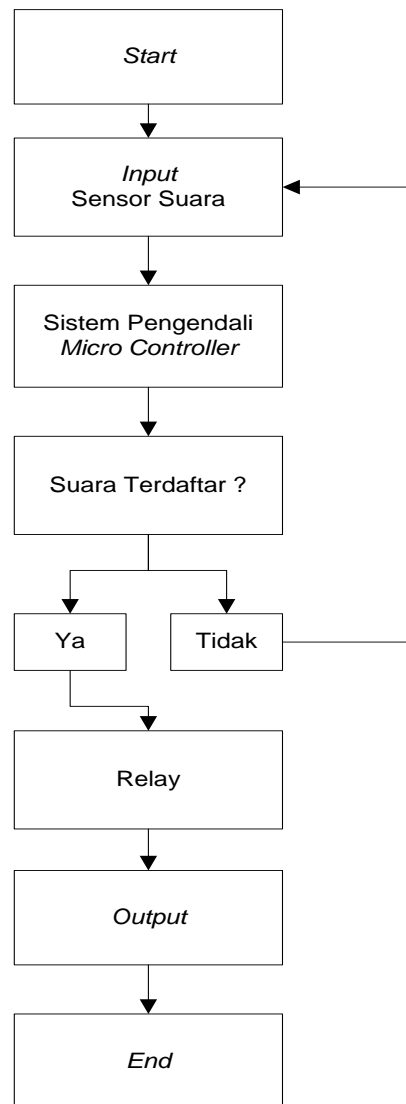
5. Relay
6. *Connector USB to TTL*
7. *Connector Serial to TTL (ISP)*
8. Laptop ASUS A46CB
9. *Stand* kelistrikan sistem penerangan (Lampu Utama dan Lampu Kecil)
serta sistem signal (*Lampu Sign* dan *Hazard*)
10. *Stand Engine Trainer*

D. Metode Penelitian

Metode penelitian yang digunakan pada penelitian adalah metode eksperimen. Dimana metode eksperimen adalah melakukan suatu percobaan dengan dilakukannya pembuatan suatu alat yang dapat mengolah suara untuk menstarter mesin dan untuk menghidupkan peralatan kendaraan (Lampu Utama, Lampu Senja, Lampu *Sign* Kanan, Lampu *Sign* Kiri dan Lampu *Hazard*). Jalannya penelitian seperti digambarkan pada alur penelitian di bawah ini:



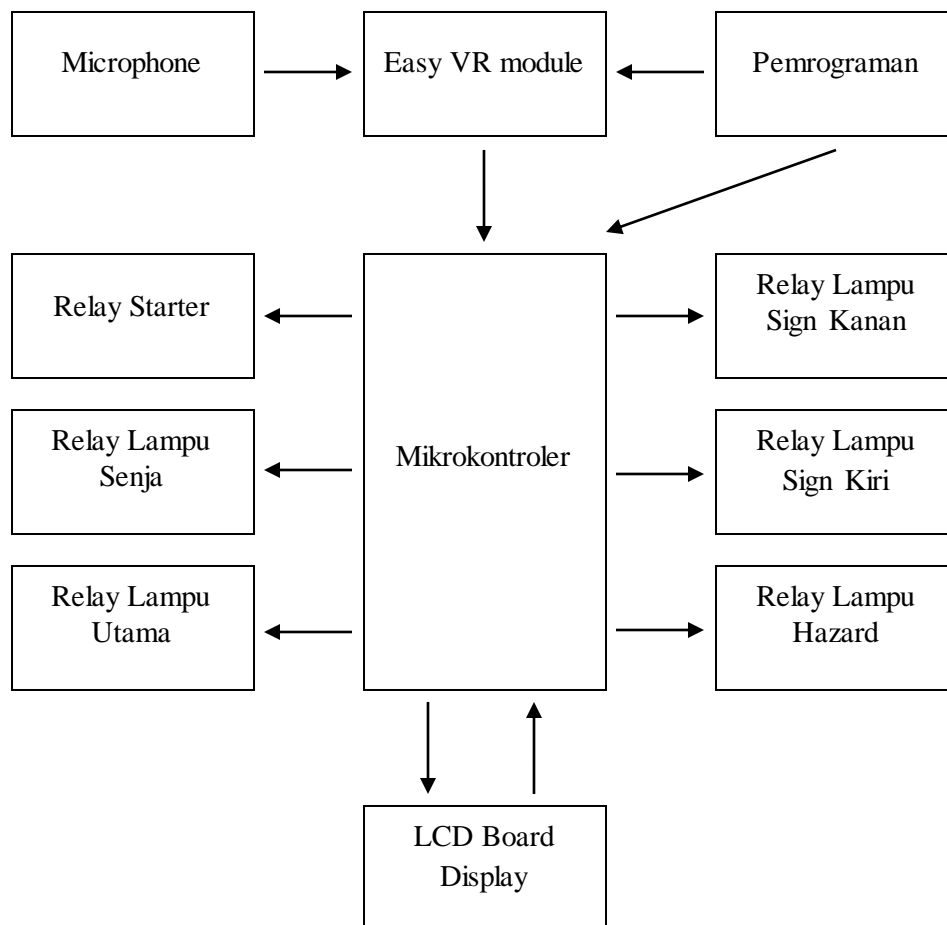
Gambar 3.1 *Flowchart* penelitian



Gambar 3.2 *Flowchart* Cara Kerja Alat Eksperimen

E. Perancangan Desain Sensor Suara

Blok diagram keseluruhan sistem yang dirancang dibagi menjadi beberapa bagian, yaitu blok modul sensor suara *EasyVR*, blok mikrokontroler utama, blok *LCD Board Display* dan blok Relay yang ditunjukkan dalam gambar 3.3.



Gambar 3.3 Diagram Blok Sistem

Fungsi masing – masing blok dalam gambar 3.3 adalah sebagai berikut :

1.) Blok *Microphone*

Terdapat *microphone* sebagai penangkap atau penerima sinyal suara yang selanjutnya akan diteruskan ke *EasyVR* Modul.

2.) Blok Modul Sensor Suara *EasyVR*

Pengendalian dilakukan melalui suara yang dibantu dengan modul *EasyVR* yang berfungsi sebagai sensor suara. Data – data suara akan di *Training* melalui komputer / laptop menggunakan aplikasi *EasyVR Commander ver.3.9.1*. Data – data yang sudah *disampling* akan dimasukkan kembali ke *database EasyVR*.

3.) Blok Mikrokontroler

Mikrokontroler yang digunakan adalah ATmega 8535 yang berfungsi untuk mengolah data dari modul *EasyVR* dan mengakses relay untuk mengendalikan pengontrol peralatan kendaraan (sistem starter, sistem penerangan dan sistem signal).

4.) Blok *LCD Board*

LCD Board yang digunakan yaitu sebagai variasi atau pemanis pada rancang bangun alat ini selain itu dapat berguna sebagai konfirmasi pengucapan.

5.) Blok Relay

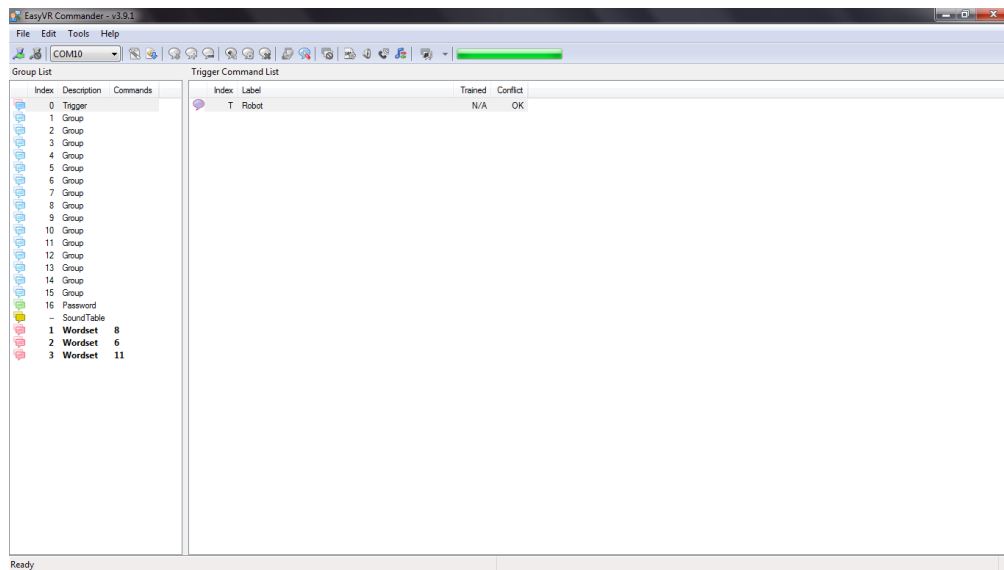
Relay digunakan sebagai saklar elektrik yang akan dilakukan percobaan terhadap 3 sistem pada kendaraan, yaitu pada sistem *starter*, sistem penerangan dan sistem *signal*.

F. Tahapan Pembuatan Alat

a. Memprogram Sensor Suara *EasyVR*

Tahapan awal dalam pekerjaan ini adalah men – *training* sensor suara *EasyVR*, pada proses *training* bertujuan untuk mengambil *sample*

suara yang akan disimpan didalam modul *EasyVR*. Pengambilan *sample* suara dilakukan dengan cara menghubungkan sensor *EasyVR* ke Laptop ASUS A46CB melalui *connector USB to TTL* serta dengan *software* yaitu *EasyVR Commander*.

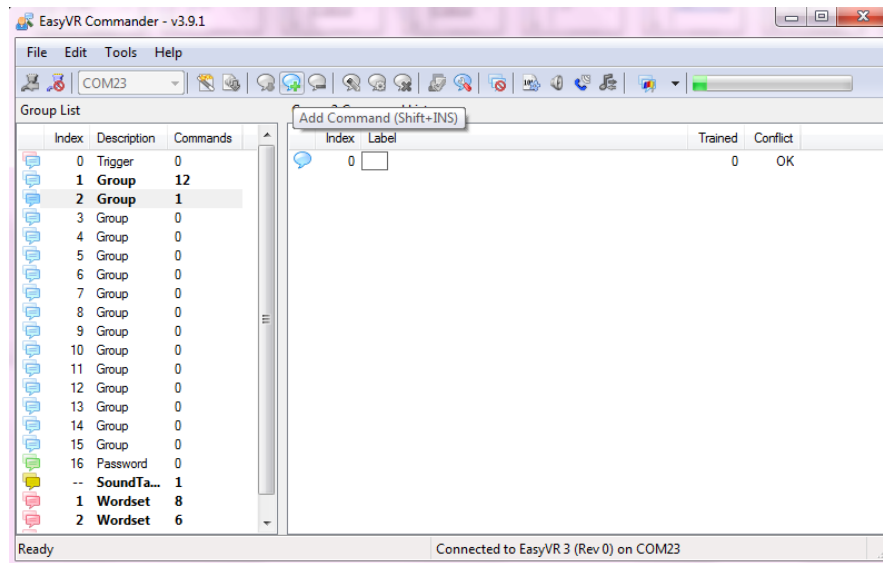


Gambar 3.4 Tampilan Utama *Software EasyVR Commander*

EasyVR Commander sendiri merupakan *software original* yang telah disediakan dari *EasyVR* yang dapat di *download* dengan gratis pada website resmi.²⁸

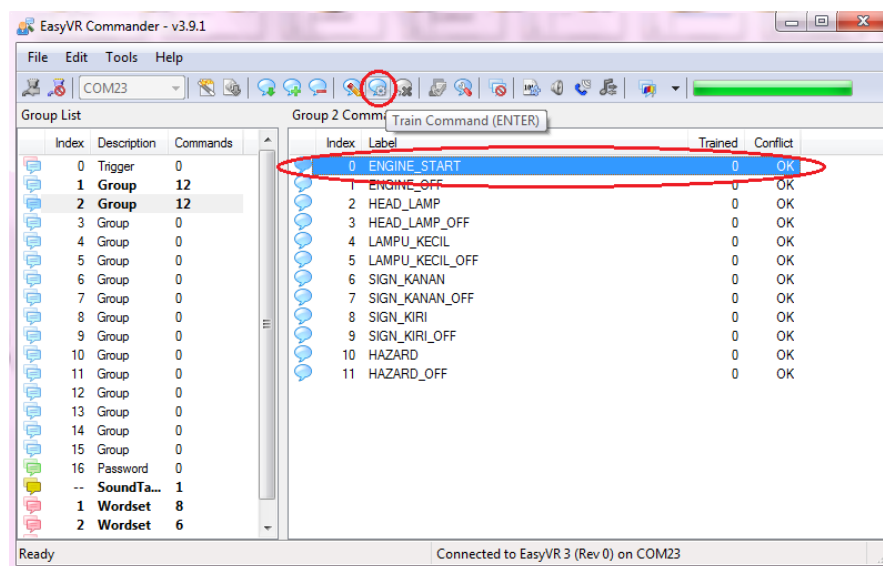
Setelah masuk kedalam *software EasyVR Commander* Langkah selanjutnya dalam pemrograman sensor ini adalah dengan cara menambahkan perintah pada sensor suara *EasyVR (Add Command)*. *Input* perintah suara sebanyak yang kita butuhkan seperti gambar 3.5.

²⁸ <http://www.veear.eu/downloads/> (diakses pada tanggal 18 mei 2015)



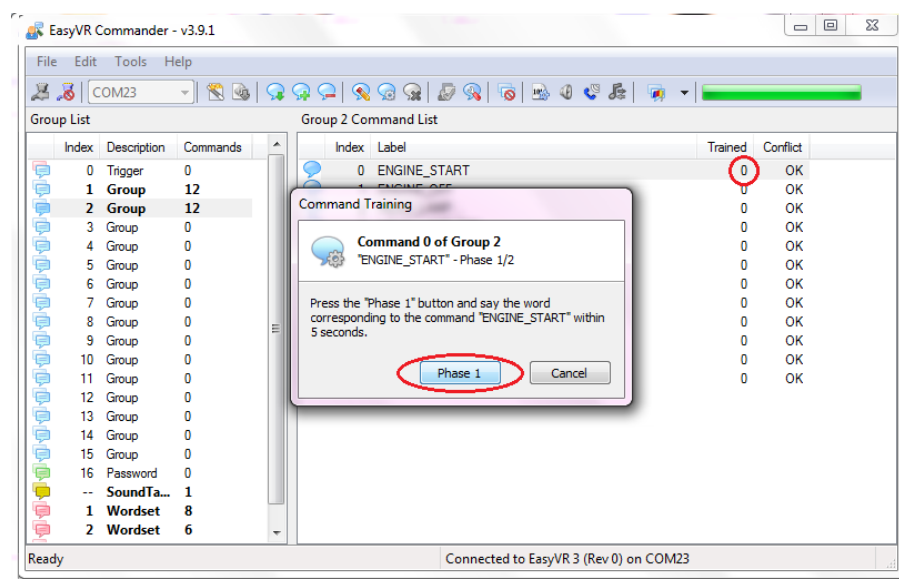
Gambar 3.5 Menambahkan Perintah (*Add Command*)

Setelah menambahkan perintah (*Add Command*) kemudian *recording* atau memberikan *Input* suara (*Train Command*) dengan cara meng klik label yang sebelumnya kita buat kemudian klik tombol *Train Command* seperti gambar 3.6.

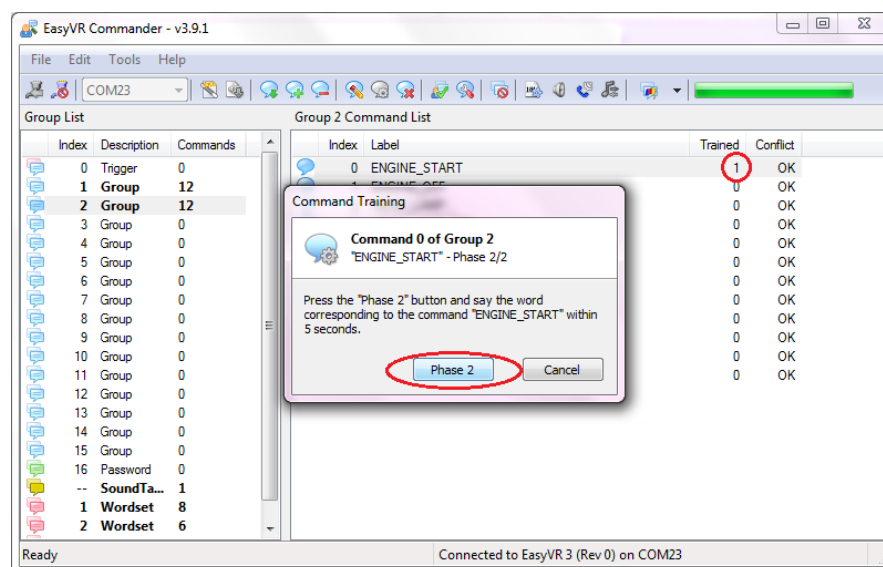


Gambar 3.6 *Recording*/memberikan input suara (*Train Command*)

Dalam proses *recording* akan memerlukan 2 kali penginputan suara, yang pertama sebagai suara pertama dan disimpan sebagai suara utama dan yang kedua merupakan konfirmasi terhadap suara yang pertama kali di input sebelumnya seperti gambar 3.7 dan 3.8.

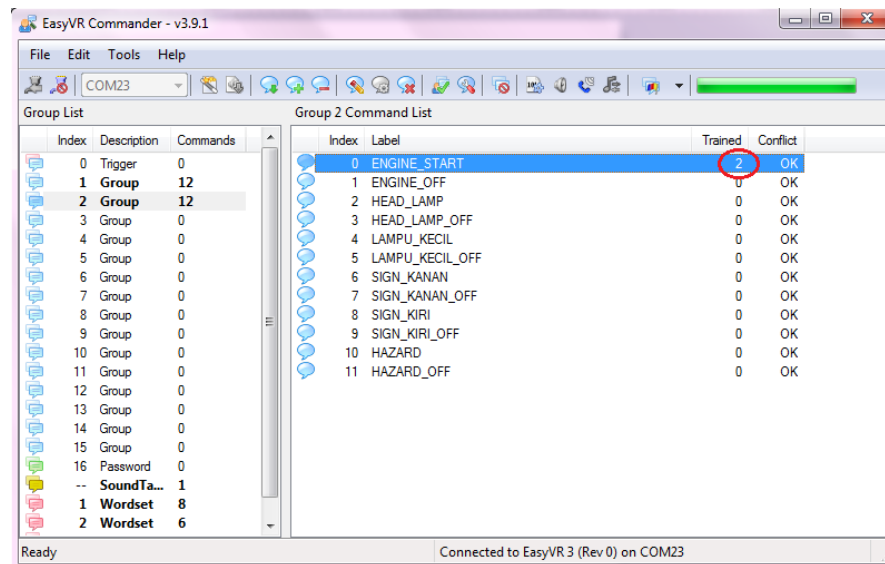


Gambar 3.7 Training Command Phase 1

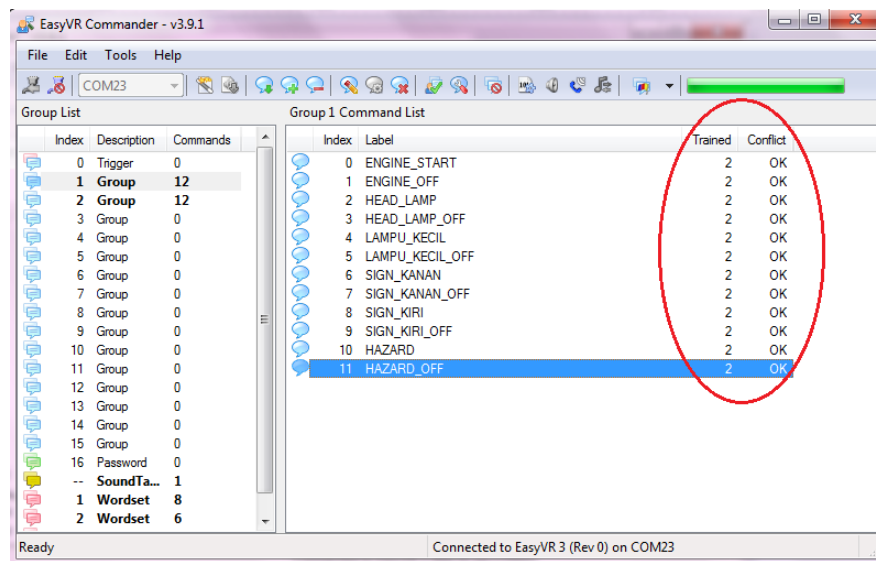


Gambar 3.8 Training Command Phase 2

Suara yang sudah terdaftar akan masuk ke *database* sensor, *Trained 2* dan *Conflict OK*, maksud dari pernyataan tersebut adalah *Input* Suara sudah terlatih 2 kali dan tidak ada konflik atau suara yang sama antara perintah 1 dengan perintah yang lainnya. Lakukan proses *trainning command* ini ke semua label yang sebelumnya kita buat (*Add Command*) hingga selesai. Seluruh *Trainning Command* yang berhasil yaitu seluruh label *Trained 2* dan *Conflict OK* Untuk Lebih jelasnya dapat dilihat pada gambar 3.9 dan 3.10.



Gambar 3.9 *Train Command* Suara Terdaftar



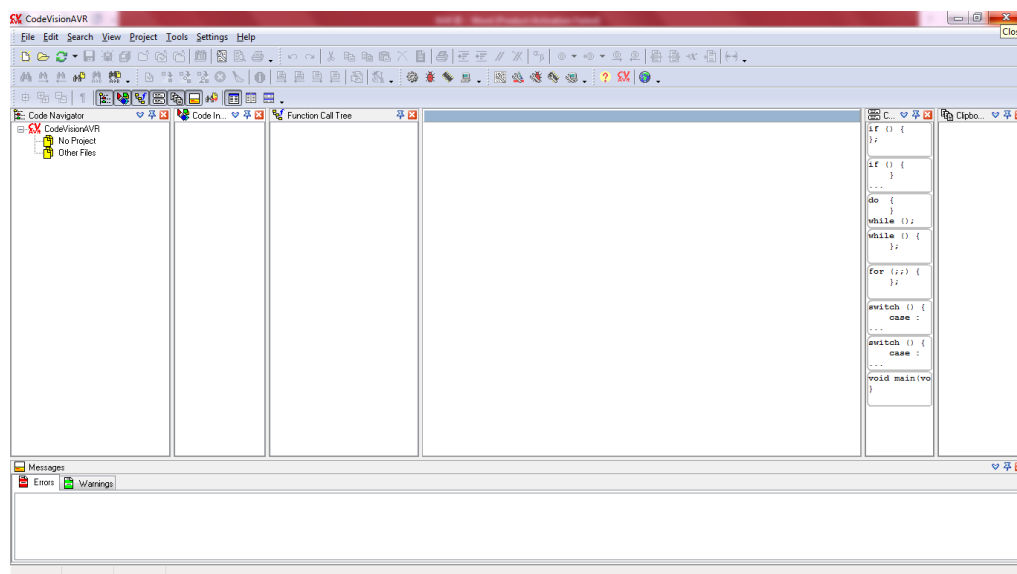
Gambar 3.10 Train Command Seluruh Perintah Suara terdaftar

Sample suara yang akan digunakan adalah sebanyak sebelas kata dengan perintah yang berbeda. Pengambilan *sample* suara dilakukan sebanyak dua kali dengan kondisi ideal atau tidak adanya *noise*, variasi kata dan intonasi disetiap pengucapan relatif sama menghasilkan kesuksesan tinggi dalam pengambilan *sample* dengan tidak adanya *error*. Kegagalan menerima variasi suara kedua dengan variasi pengucapan suara pertama akan menimbulkan kegagalan. Oleh karena itu, agar *EasyVR* dapat berfungsi dengan baik, dibutuhkan variasi suara yang relatif sama dengan *sample*.

b. Memprogram Mikrokontroler

Tahapan proses yang terdapat pada sistem ini meliputi proses pengolahan data dari modul *EasyVR* ke Mikrokontroler dan proses pengontrolan *relay*. Semua proses tersebut dilakukan oleh perangkat

lunak yang terdapat dalam mikrokontroler. Perangkat lunak ini tersusun dari instruksi-instruksi yang membentuk sebuah *listing* program atau *source code*. Semua instruksi program disusun secara terstruktur dalam beberapa subrutin yang secara khusus menangani fungsi tertentu. Dalam pemrograman mikrokontroler dibuat menggunakan *software Code Vision AVR* (CV AVR). Menggunakan bahasa pemrograman yaitu bahasa pemrograman C.



Gambar 3.11 Tampilan Utama *Software CV AVR*

CV AVR merupakan salah satu dari beberapa *software* yang berfungsi sebagai pemrograman mikrokontroler. Dalam proses pekerjaan ini yang dilakukan yaitu memprogram penulisan pada *LCD Board* dan mengatur ke aktifan relay dengan perintah serta memberikan jeda waktu terhadap kebutuhan relay tertentu.

```

D:\Script Sweet\BANCANG BANGUN PENGONTROL PERALATAN KENDARAAN DENGAN SENSOR SUARA\all about this script\Aplikasi_Pemrograman\PROGRAM\2.c
Notes 2.c
37 unsigned char y;
38 putchar(' ');
39 y=getchar();
40 if(y=='A'){lcd_gotoxy(0,1);lcd_putsf("ENGINE START ");PORTC.1=1;delay_ms(1000);PORTC.0=1;delay_ms(3000);PORTC.0=0;}
41 if(y=='B'){lcd_gotoxy(0,1);lcd_putsf("ENGINE OFF ");PORTC.1=0;} //Relay 2 OFF KONTAK MESIN
42 if(y=='C'){lcd_gotoxy(0,1);lcd_putsf("HEAD LAMP ON ");PORTC.2=1;} //Relay 3 ON
43 if(y=='D'){lcd_gotoxy(0,1);lcd_putsf("HEAD LAMP OFF ");PORTC.2=0;} //Relay 3 OFF
44 if(y=='E'){lcd_gotoxy(0,1);lcd_putsf("LAMPU KECIL ");PORTC.3=1;} //Relay 4 ON
45 if(y=='F'){lcd_gotoxy(0,1);lcd_putsf("LAMPU KECIL OFF ");PORTC.3=0;} //Relay 4 OFF
46 if(y=='G'){lcd_gotoxy(0,1);lcd_putsf("SIGN KANAN ON ");PORTC.4=1;PORTC.5=0;} //Relay 5 ON
47 if(y=='H'){lcd_gotoxy(0,1);lcd_putsf("SIGN KANAN OFF ");PORTC.4=0;} //Relay 5 OFF
48 if(y=='I'){lcd_gotoxy(0,1);lcd_putsf("SIGN KIRI ON ");PORTC.5=1;PORTC.4=0;} //Relay 6 OFF
49 if(y=='J'){lcd_gotoxy(0,1);lcd_putsf("SIGN KIRI OFF ");PORTC.5=0;} //Relay 6 OFF
50 if(y=='K'){lcd_gotoxy(0,1);lcd_putsf("HAZARD ");PORTC.4=1;PORTC.5=1;} //Relay 5,6 HAZARD
51 if(y=='L'){lcd_gotoxy(0,1);lcd_putsf("HAZARD OFF ");PORTC.4=0;PORTC.5=0;} //Relay 5,6 HAZARD
52
53 return;
54 }
55
56 void voice_start()
57 {
58 unsigned char x;
59 lcd_gotoxy(0,1);lcd_putsf("SPEAK NOW...!! ");
60 putchar('d');
61 putchar('B');
62 x=getchar();
63 if(x=='x'){lcd_gotoxy(0,1);lcd_putsf("command ok ");delay_ms(1000);action();}
64 if(x=='e'){lcd_gotoxy(0,1);lcd_putsf("command error ");return;}
65
66 }

```

Gambar 3.12 Program pada Mikrokontroler

Program pada mikrokontroler seperti pada gambar diatas dapat dijelaskan sebagai contoh bahwa jika simbol A hidup maka LCD akan hidup dan menunjukkan tulisan *ENGINE START* kemudian Relay 2 sebagai kontak (ACC) hidup lalu 1 detik kemudian Relay 1 yang menghubungkan ke *starter* akan hidup dan dalam waktu 3 detik Relay 1 akan mati dan Relay 2 akan hidup hingga *command* selanjutnya di perintahkan. Jika simbol B hidup maka LCD akan hidup dan menunjukkan tulisan *ENGINE OFF* kemudian Relay 2 sebagai kontak (ACC) akan OFF. Jika simbol C hidup maka LCD akan hidup dan menunjukkan tulisan *HEAD LAMP ON* kemudian Relay 3 sebagai kontak ke lampu utama akan ON. Jika simbol D hidup maka LCD akan hidup dan menunjukkan tulisan *HEAD LAMP OFF* kemudian Relay 3 sebagai kontak ke lampu utama akan OFF. Jika simbol E hidup maka LCD akan hidup dan menunjukkan tulisan *LAMPU KECIL* kemudian

Relay 4 sebagai kontak ke lampu kecil akan *ON*. Jika simbol F hidup maka LCD akan hidup dan menunjukkan tulisan *LAMPU KECIL OFF* kemudian Relay 4 sebagai kontak ke lampu kecil akan *OFF*. Jika simbol G hidup maka LCD akan hidup dan menunjukkan tulisan *SIGN KANAN ON* kemudian Relay 5 sebagai kontak ke *sign* kanan akan *ON*. Jika simbol H hidup maka LCD akan hidup dan menunjukkan tulisan *SIGN KANAN OFF* kemudian Relay 5 sebagai kontak ke *sign* kanan akan *OFF*. Jika simbol I hidup maka LCD akan hidup dan menunjukkan tulisan *SIGN KIRI ON* kemudian Relay 6 sebagai kontak ke *sign* kiri akan *ON*. Jika simbol J hidup maka LCD akan hidup dan menunjukkan tulisan *SIGN KIRI OFF* kemudian Relay 6 sebagai kontak ke *sign* kiri akan *OFF*. Jika simbol K hidup maka LCD akan hidup dan menunjukkan tulisan *HAZARD* kemudian Relay 5 & 6 sebagai kontak ke *hazard* akan *ON*. Jika simbol L hidup maka LCD akan hidup dan menunjukkan tulisan *HAZARD OFF* kemudian Relay 5 & 6 sebagai kontak ke *hazard* akan *OFF*.

Berikut merupakan tabel yang berisikan perintah suara, simbol pada mikrokontroler serta relay yang digunakan :

Tabel 3.1. Perintah Suara, Simbol dan Relay

NO.	PERINTAH SUARA	SIMBOL PADA MIKROKONTROLER	RELAY AKTIF
1	ENGINE START	A	1
2	ENGINE OFF	B	2
3	HEAD LAMP	C	3
4	HEAD LAMP OFF	D	3
5	LAMPU KECIL	E	4
6	LAMPU KECIL OFF	F	4
7	SIGN KANAN	G	5
8	SIGN KANAN OFF	H	5
9	SIGN KIRI	I	6
10	SIGN KIRI OFF	J	6
11	HAZARD	K	5 & 6
12	HAZARD OFF	L	5 & 6

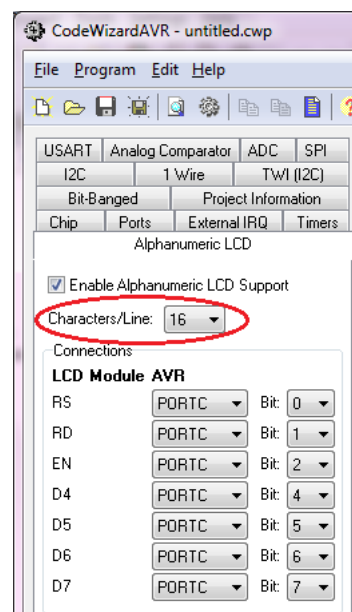
c. Memprogram LCD

Alphanumeric LCD merupakan sebuah *Liquid Crystal Display* yang terdiri dari segmen – segmen yang berfungsi untuk menampilkan karakter, baik berupa angka, simbol maupun huruf. *Display* yang akan dibahas adalah *display LCD 2 x 16*, artinya LCD memiliki 2 baris dan 16 karakter. Berikut merupakan cara pemrograman LCD 2 x 16 :

1. Buka *Code Vision AVR*
2. Buatlah *project* baru dengan cara berikut

Pilih *File – New – Project – Ok – Code Wizard Yes*

3. Lakukan pengaturan *Alphanumeric LCD*, pastikan *character/line* adalah 16 karena dalam pembuatan alat ini menggunakan LCD 2 x 16 untuk lebih jelasnya dapat dilihat pada gambar dibawah 3.13 dibawah ini.

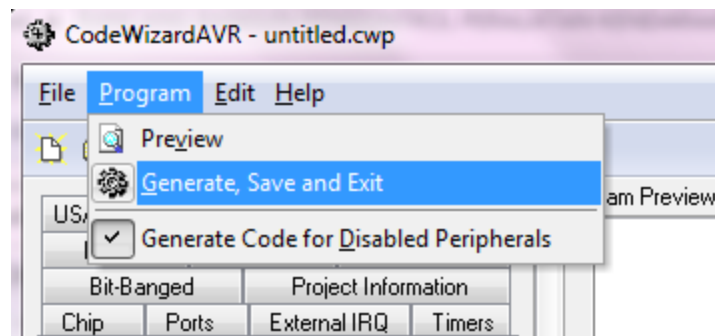


Gambar 3.13 Pengaturan awal *Alphanumeric LCD*

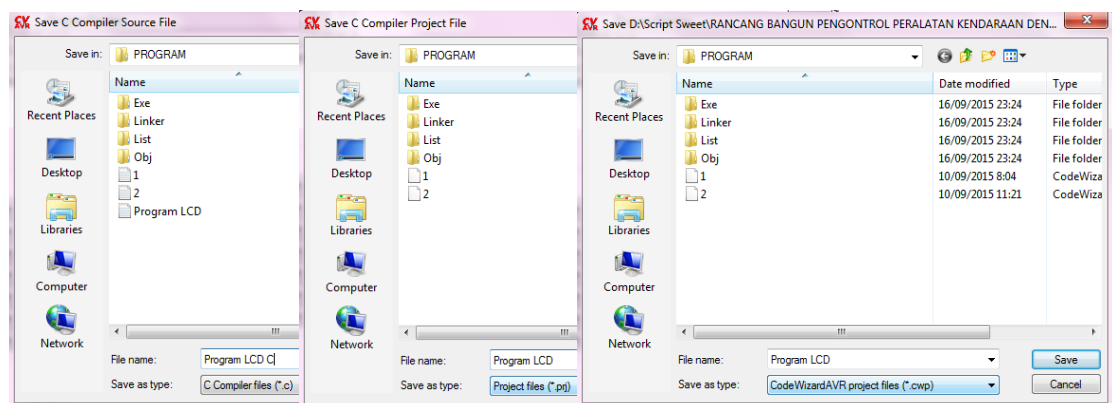
4. Simpan *project* pada direktori yang diinginkan.

Pilih *File – generate save and exit*

Dalam penyimpanan ini akan menyimpan 3 file berbeda agar lebih mempermudah simpan semua *file* dengan nama yang sama.

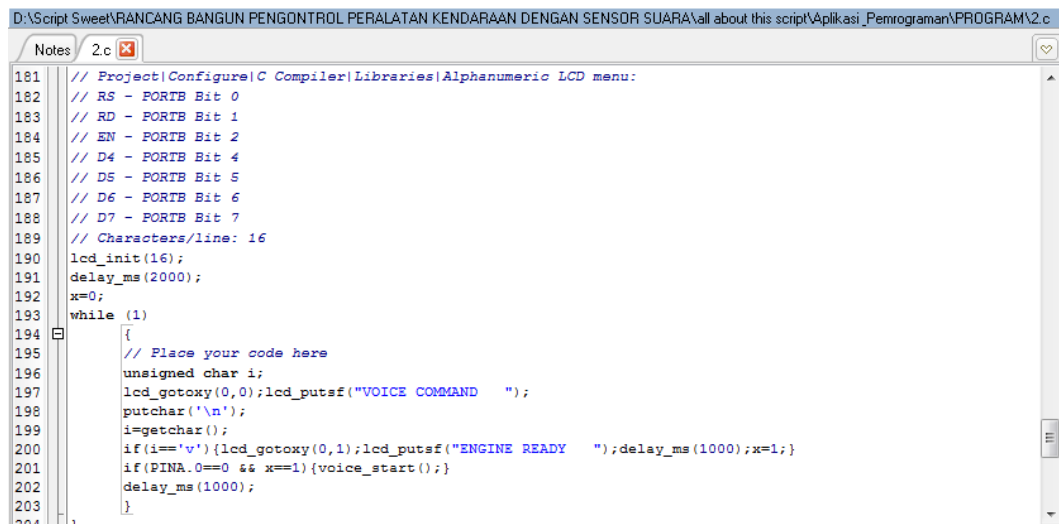


Gambar 3.14 Penyimpanan File



Gambar 3.15 Penyimpanan 3 File

5. Ketikkan program berikut



```

181 // Project\Configure\C Compiler\Libraries\Alphanumeric LCD menu:
182 // RS - PORTB Bit 0
183 // RD - PORTB Bit 1
184 // EN - PORTB Bit 2
185 // D4 - PORTB Bit 4
186 // D5 - PORTB Bit 5
187 // D6 - PORTB Bit 6
188 // D7 - PORTB Bit 7
189 // Characters/line: 16
190 lcd_init(16);
191 delay_ms(2000);
192 x=0;
193 while (1)
194 {
195     // Place your code here
196     unsigned char i;
197     lcd_gotoxy(0,0);lcd_putsf("VOICE COMMAND  ");
198     putchar('\n');
199     i=getchar();
200     if(i=='v'){lcd_gotoxy(0,1);lcd_putsf("ENGINE READY  ");delay_ms(1000);x=1;}
201     if(PINA.0==0 && x==1){voice_start();}
202     delay_ms(1000);
203 }

```

Gambar 3.16 Pemrograman LCD menggunakan CV AVR

6. Lakukan kompilasi

Pilih *Project – build all*

Jika terjadi kesalahan penulisan program maka *software* akan menampilkan pesan *error*. Perbaiki program tersebut.

7. Simpan *project*

d. Pembuatan *Trainer* sistem penerangan dan sistem *signal*

Trainer ini berfungsi sebagaiudukan atau tempat untuk seluruh rangkaian baik sumber arus (baterai), sensor suara, mikrokontroler, relay serta *LCD Board*. Selain itu *output* atau beban seperti lampu utama, lampu kecil, serta lampu *sign* juga ada di *trainer* ini. Dalam pembuatan *trainer* ini menggunakan bahan besi siku dengan ketebalan

3mm dan akrilik dengan ukuran 35 cm x 20 cm dengan ketebalan 5 mm. Berikut merupakan langkah kerja dari pembuatan *trainer* kelistrikan sistem *signal* dan sistem penerangan :

1. Plat siku yang akan digunakan sebagai kerangka di potong sesuai dengan kebutuhan yaitu dudukan bawah 35cm x 20cm, dengan ketinggian 100cm dan *holder* akrilik 35cm x 20cm dengan ketebalan plat siku 3mm, setelah dipotong kemudian di sambungkan dengan cara dilas. Setelah itu dilanjutkan dengan pemasangan roda pada setiap sisi dudukan.



Gambar 3.17 Besi Siku yang Sudah di Potong dan di Las

2. Melakukan penggerindaan di setiap bagian yang telah dilas. Penggerindaan paling banyak dilakukan pada bagian sudut kerangka. Hal ini dilakukan agar bagian hasil pengelasan terlihat lebih rapi.
3. Melakukan proses pendempulan pada setiap bagian yang telah digerinda.



Gambar 3.18 Stand Kelistrikan Setelah di Las

5. Melakukan proses pengecatan dasar *trainer* kelistrikan.
6. Mengecat dan melakukan *finishing* pada *trainer* kelistrikan.
7. Memasang lampu utama dan lampu kecil pada *trainer*.
8. Memasang lampu *sign* kanan dan kiri pada *trainer*.
9. Memasang akrilik pada *trainer*.
10. Memasang sensor suara pada akrilik.
11. Memasang mikrokontroler pada *DI-Smart AVR 16 System*.
12. Memasang *DI-Smart AVR 16 System* pada akrilik.
13. Memasang *LCD board* pada akrilik.
14. Memasang *relay board* pada akrilik.
15. Merangkai seluruh komponen semua sistem.